Construction of Stochastic Simulation Metamodels Using *Splines*

Rui Barradas
Número: 68239
Email: ruihsb@hotmail.com

Instituto Superior Técnico, Lisbon, Portugal

Abstract. The simulation models of real systems are employed in order to provide support for decisions concerning about possible alterations in the design of these systems. A metamodel is an analysis tool that relates the input data of the simulation model with its results. The polynomial metamodels are easier to use, since they rely on a single polynomial function. However, these metamodels based on polynomials have some difficulties when it comes to adjusting to arbitrary regular functions and to the non-linearity of the problems of the real-life. In this work, it is made the construction of alternative metamodels, based in splines, which seek to overcome some of these limitations. Furthermore, it is made a comparison between these metamodels based in splines and the polynomial metamodels, in order to check if the first ones allow to obtain better adjustments.

Keywords: Stochastic Simulation; Simulation Models; Metamodel; Methodology; *Splines*

1 Introduction

Simulation models are used as a tool for exploring the real systems. These models can be used to simulate situations that are extremely difficult or expensive to test in the real life. The stochastic simulation is an analysis technique of stochastic models. These models have one or more random variables as input parameters, reason why they generate random outputs.
The metamodels relate the input values with the response of the simulation model by a mathematical function.
Within the set of the several metamodels, the polynomial metamodels have received special consideration due to their simplicity. However, the polynomials do not have the ability to adjust to arbitrary shape functions based on values that can be disturbed by stochastic components with overall distribution.
Furthermore, most of the real life problems are non-linear. Thus, in certain cases, linear approximations may be inappropriate to use polynomial metamodels to perform the adjustments.
In this work, we look up to study a methodology that consists in the construction of metamodels using *splines*. A *spline* is a numeric function that is based in

polynomials, defined by a set of intervals where each interval corresponds to a polynomial.
These functions allow the construction of more complex metamodels than those possibly by a single polynomial. Thus, we will seek to use *spline* functions to see if they can cover some of the deficiencies of the single polynomial metamodels.

2 Background

2.1 Stochastic Simulation

The stochastic simulation is a review mechanism to evaluate non-deterministic models. These models allow the representation of random variables and their values. Due to these variables, whenever a non-deterministic model is evaluated, even if the given parameters remain, its behavior may change [2].
Once this work fits into the context of stochastic simulation, it is important to mention the major advantages and disadvantages of using stochastic. The most complex real-world systems with statistical elements are too complex to be described in detail by a mathematical model that can be evaluated analytically. As a result, the simulation is generally the only possible type of investigation. Furthermore, the use of simulation allows to check the performance of a system over a given set of operating conditions. In a simulation, it's possible to have a better control over the experimental conditions than that which would be possible when undertaking experiments directly in the real system [4].
On the other hand, the main disadvantage is that every run of a stochastic simulation model generates a single value which evaluates the real system characteristics under a particular set of input data. The stochastic nature of the system's behaviour requires that several independent runs for each set of input data in order to obtain more accurate estimates of the quantities on study.

2.2 Output Modeling

Before evaluating the results of a simulation study, it is important to take into account the properties of the stochastic simulation and the need for a proper methodology for the analysis of the results. Once the simulation model is constructed, it should be used appropriate statistical techniques to analyse the results.
In the stochastic simulation models several problems must be taken into account, depending on the type of simulation that we are running.
The simulations can be distinguished in two types: terminating or non-terminating. The difference between the two types depends on whether we are interested in the behavior of the system during a given period of time (terminating) or in the stationary behavior of the system (non-terminating).

2.2.1 Initialization bias In some systems, such as the queueing systems, when the process starts, there is an initial transient behaviour, since the simulation program starts without clients waiting to be served.
Whenever the aim of study is the behavior of system in a steady state, it is necessary to allow some time for the system to stabilize. Therefore, initial results are biased and should not be used in the analysis, since they may not represent the reality of the system in stationary phase.
These initial observations should be eliminated or ignored, during the statistical analysis. The set of initial observations of this series is referred to as an initialization bias. The initialization bias can be considered as the major source of error in the estimation of the value of the measure of the system performance in the non-terminating simulations.
There are various methods for the detection of the initialization bias, among which we highlight the graphical methods and statistical methods. The method used in this work was the statistical Permutations Test method, developed by Enver Yücesan [8].

2.2.2 Samples In a stochastic simulation, since it depends on random variables, a single replica of the simulation is not conclusive, as the resulting value from the simulation can be quite far from the real mean value.
There are two important methods to make a proper analysis of the results: the Independent Replicas method and the Batch-Means method.

2.2.2.1 **Independent Replicas** In this method, the simulation is repeated a certain number of times, each time using a different sequence of independent pseudo-random numbers. Then, the performance measure (e.g., mean) of the observations collected is determined for each execution of the simulation program. In this way, the values are independent and identically distributed. The end result of the simulation is obtained by calculating the average of these values [5].

2.2.2.2 **Batch-Means** This method consists in removing a certain number of observations from a single simulation, and divide that number into batches. The size of each batch must be determined in order to ensure that the observation in each batch are independent up to a given significance level. Then, it is calculated the performance measure of the observations which form each batch, and finally, the average value of these measures is determined for each batch [1].

2.3 Simulation Metamodels

Nowadays, it is extremely important to evaluate the performance of a real system. Since the real system may not have the desired performance, it may be necessary to make some changes. However, make these changes in the real system can be complex and expensive to perform. So, it is useful to create a simulation model that accurately corresponds to the real system under study. The resulting simulation model is used to test the performance and the behavior of the

real system for a range of input data. In order to evaluate the performance of a measure of interest in the system, it is necessary to evaluate the input data and the results, and understand the relationship between them. The construction of metamodels, which relate the behavior of the input data of the simulation model and its results by a mathematical function, provides a simple and clear representation for the measure of interest [7].

2.3.1 Polynomial Metamodels The polynomial metamodels use a single polynomial function to approximate the response in the entire domain of interest. The polynomials obey to the following general form:

$$f(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + ... + a_nx^n. \quad (1)$$

The degree of the polynomial is determined by the highest coefficient which is different from zero. For example, if a_2 is different from zero, but the coefficients a_3 and above are equal to zero, then the polynomial is of degree 2.
The use of polynomials in the construction of metamodels has some disadvantages, due to their simplicity. The use of polynomial metamodels are good when used in linear problems. However, in real life, non-linearity problems arise frequently. Therefore, approximations using these metamodels become, sometimes, unrealistic.

2.3.2 Metamodels based in Splines A spline function is a mathematical function defined by intervals, delimited by a sequence of points, wherein each interval corresponds to a polynomial. Let it be $a = x_1 < x_2 < x_3 < ... < x_n = b$ one ordered set of points (or nodes), contained in any interval $[a; b]$. A spline function with order $(m + 1)$ is a function piecewise defined by polynomials of degree m, with $m - 1$ continuous derivatives at the nodes [6].

2.3.2.1 **Interpolating Splines** A spline function that passes through the control points is called an interpolating spline. These functions can acquire the most diverse ways, always according to the type of spline that we are using. The type of a spline depends on the degree of the polynomials that compose it (linear, quadratic, cubic, etc.). The interpolating splines allow for good approximations if they are built on sufficiently accurate results [3].

2.3.2.2 **Smoothing Splines** A spline function that passes near the control points, but not necessarily through them, is called a smoothing spline. The motivation behind a smoothing spline is based on the relationship between the accuracy of the approximations to the points and the smooth of the resulting model. These functions are adjusted by a parameter (λ), which controls the regularization of the spline; this is called as smoothing parameter. In this work, this kind of splines was used to perform the adjustments.

3 Implementation

3.1 Systems used in the study

There were used two systems during the realization of the work: the M/M/1 queueing system and a routing network.
The M/M/1 system models the arrival rate to the queue based on a Poisson process to an unlimited size queue and the service rate for a single server has a fixed mean exponential distribution. The number of users is unlimited in order to determine steady state measures of performance. Theoretical values for most measures of interest are available and can be used to assess the accuracy of the constructed metamodels.
In the routing network, the messages arrive via a network with three interconnected queues with limited capacity. Each message requires a fixed processing time, followed by a separation process which reprocesses some of the messages.

3.2 Analysis

In order to perform the output analysis for both systems the following steps were taken:

1. Simulate the respective system in the AweSim program for each input value in the design region;
2. Use the Matlab program to perform the treatment of the results for each of the resulting input values:
   (a) Remove the initialization bias of the results using the Permutations Test method;
   (b) Use the Batch-Means method in order to determine the required performance measures: mean and variance values for the time that an entity expends on the system.

The initial bias and the size of each batch of each response were determined using the Permutations Test method developed in Matlab. The function consisting of:

```
function [ out, rho, ms, sl, split ] = yucesan(y, b, m, corr, NS, NL);
```

where y is the vector of the collected observations (the size should be at least b $\times$ m), b is the initial number of batches, m is the initial size of the batches, *corr* is the value of the maximum correlation allowed between batches, *NS* is the number of shuffles and *NL* is the desired level of significance. The result of this function describes the number of observations which must be presented in each batch (ms), as well as the number of observations that correspond to the initialization bias ($out = split \times ms$), the achieved maximal serial correlation among batches (rho), the lowest significance level obtained (sl) and the number of batches to be removed ($split$).
The mean and variance values were determined using a Matlab function, *meanstd*, which receives three parameters: the text file containing the time in system values for each entity, the number of observations in each batch and the number of

observations of the initialization bias. Based on these parameters, the function removes the observations corresponding to the initialization bias and calculates the number available of batches. With these values, it was possible to calculate the mean of each batch and then the mean and variance of these means.

3.2.1 M/M/1 Queueing System The simulation of the M/M/1 queueing system in the AweSim program results in a set of files, wherein each of these files corresponds to a different value of the utilization factor ($\rho = \frac{\lambda}{\mu} = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95\}$). Each of these files contains the recorded time spent in the system values for the different entities.
The resulting mean and variance values after the use of the *yucesan* and *meanstd* functions are presented in the table 1.

ρ	*bias*	*batchSize*	Number of batches (*numberBatches*)	$\bar{x}$	$\bar{\sigma}_x$	$\bar{\sigma}_x^2$
0.1	450	225	86	0.1110	0.0084	7.1328^{-5}
0.2	725	725	26	0.2507	0.0125	1.5749^{-4}
0.3	650	50	387	0.4310	0.1130	0.0128
0.4	350	350	56	0.6607	0.0644	0.0041
0.5	6250	250	55	1.0402	0.2184	0.0477
0.6	3000	150	113	1.5354	0.4509	0.2033
0.7	3900	260	62	2.3208	0.7471	0.5581
0.75	4000	160	100	2.9453	1.4116	1.9925
0.8	4750	250	61	4.2364	2.0995	4.4078
0.85	7475	325	38	6.0117	3.5191	12.3842
0.9	1610	115	159	8.8992	7.1232	50.7399
0.95	5240	1310	11	17.5029	9.3312	87.0711

Table 1. Results of the *yucesan* and *meanstd* functions to the M/M/1 queueing system: $\bar{x}$ = mean of the batch-means and $\bar{\sigma}_x^2$ = variance of the batch-means.

Note that these values represent the mean time of the entities in the system. Therefore, the values shown in the table 1 refer to the mean and variance of the mean time values recorded in the system for the various entities, for the various values of ρ. Table 1 shows that the mean time that an entity expends in the system grows as the utilization factor increases. This behavior holds for the all values of ρ.

3.2.2 Routing Network The simulation of the routing network in the AweSim program results in a set of files, one for each value of the arrival rate (λ). As for the M/M/1 queueing system, each of these files contains the recorded time in system values for the different system entities. The experimental input data points $\lambda = \{\frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 2^{-0.5}, 1, 2^{0.5}, 2, 4, 8, 16, 32\}$ were used.
The final results after the use of the *yucesan* and *meanstd* functions are presented in the table 2.

λ	*balk*	*bias*	*batchSize*	*numberBatches*	$\bar{x}$	$\bar{\sigma}_x$	$\bar{\sigma}_x^2$
1/32	0	6750	150	288	13.4179	0.4101	0.1682
0.0625	0	120	40	1247	13.4453	0.7274	0.5291
0.125	0	460	230	215	13.5073	0.3312	0.1097
0.25	0	1050	150	326	13.6619	0.4626	0.2140
0.5	0	3360	140	333	14.5726	0.6693	0.4479
$2^{-0.5}$	11	550	550	89	16.5759	0.9292	0.8634
1	2131	2480	1240	38	26.6885	2.3062	5.3187
$2^{0.5}$	21522	440	440	112	36.1317	1.8680	3.4893
2	49468	1050	350	139	37.6709	2.3442	5.4955
4	151175	6360	530	82	38.4238	1.9422	3.7720
8	350886	3770	130	355	38.4848	3.4589	11.9639
16	745746	50	50	999	38.3222	3.8397	14.7429
32	1560785	960	320	153	38.0706	2.1563	4.6495

Table 2. Results of the *yucesan* and *meanstd* functions to the routing network: $\bar{x}$ = mean of the batch-means ; $\bar{\sigma}_x^2$ = variance of the batch-means ; *balk* = number of lost packets in the system.

The analysis of the table 2 allows us to conclude that:

- At an early stage ($1/32 \leq \lambda \leq 0.5$), while there was no packet losses on the network, the registered average times were of the same order ($\bar{x} \in [13.4179; 14.5726]$). Hence, the network is not congested, so the average time in system was low;
- At an intermediary stage, as soon as packet loss began to occur, the average time increased as the arrival rate also increased. This is visible for $\lambda \in [2^{-0.5}; 2]$. This means that the network becomes more and more saturated and an increase in the average time in system is recorded, and the number of packet losses on the network increases.
- At the final stage ($4 \leq \lambda \leq 32$), the network reaches its full capacity and the highest average time in system were recorded. The average time in system stabilizes at values of the same order ($\bar{x} \in [38.0706; 38.4848]$), but the number of packet losses increases significantly.

4 Experimental results

The estimation of the metamodel functions, using polynomial and spline approximations, are performed using existing Matlab curve fitting routines. For each estimation a set of values are obtained for the metamodel's unknown parameters and for the resulting sum of square error ($\mathrm{SSE_e}$). The $\mathrm{SSE_e}$ is used as measure of the metamodel's accuracy in order to establish a comparison between the two types of metamodels. For the M/M/1 queueing system, the metamodel's error in relation to the theoretical values ($\mathrm{SSE_t}$) can be computed since theoretical values are known.

Two sets of metamodels were constructed for each system, for polynomial and spline fittings. Since the variance is unequal along the experimental region, a set of metamodels is computed using the weighted mean time in system. The other set assumes equal variance for the mean values. The weight calculation at each design point i is calculated using the previously determined variances:

$$w[i] = \sqrt{\frac{1}{\text{var}[i]}} = \frac{1}{\text{std}[i]}, \tag{2}$$

where $\text{var}[i]$ is the variance of the point i and $\text{std}[i]$ is the respective standard-deviation.

4.1 M/M/1 Queueing System

4.1.1 Adjustments based on polynomial metamodels The results of the errors for the M/M/1 Queueing system calculated from the adjustments based on polynomial metamodels were present in the table 3.

Degree	SSE_t without weights	SSE_e without weights	SSE_t with weights	SSE_e with weights
1	124.8318	124.5240	348.4997	348.1924
2	51.4028	50.6309	247.5927	246.8116
3	18.6588	17.2957	134.1107	129.4528
4	7.8399	5.9835	50.3805	54.9238

Table 3. Errors calculated for the polynomial metamodels for the M/M/1 queueing system.

4.1.2 Adjustments based on spline metamodels The results of the errors for the M/M/1 queueing system calculated from the adjustments based on spline metamodels were present in the table 4.

Smoothing parameter (P)	SSE_t without weights	SSE_e without weights	SSE_t with weights	SSE_t with weights
0.999	16.1222	18.7992	49.9205	53.9190
0.99	49.3052	52.6680	118.0594	120.0427
0.97	76.2357	78.8713	156.9692	157.9604
0.9	104.4926	105.5828	204.4753	204.6669

Table 4. Errors calculated for the spline metamodels for the M/M/1 queueing system.

4.2 Routing Network

4.2.1 Adjustments based on polynomial metamodels The results of the errors for the routing network calculated from the adjustments based on polynomial metamodels were present in the table 5.

Degree	SSE without weights	SSE with weights
1	383.1446	1057.4144
2	382.5750	666.4289
3	168.5915	351.7482
4	168.5350	322.8838

Table 5. Errors calculated for the polynomial metamodels for the routing network.

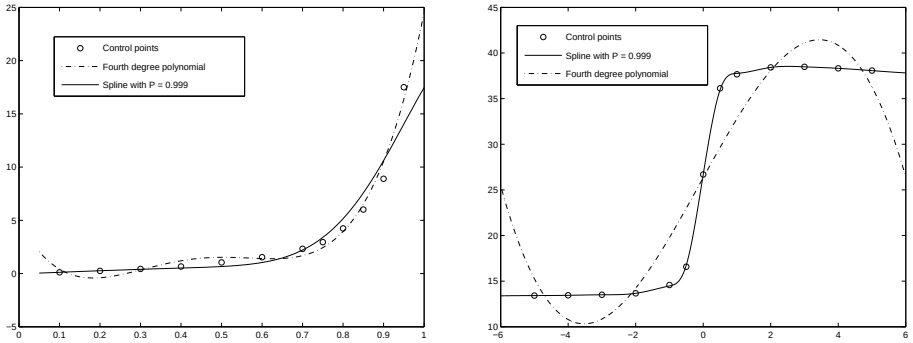
4.2.2 Adjustments based on spline metamodels The results of the errors for the routing network calculated from the adjustments based on spline metamodels were present in the table 6.

Smoothing parameter (P)	SSE without weights	SSE with weights
0.999	0.0878	0.1711
0.9	20.9633	29.4727
0.8	37.2324	52.1113
0.4	104.6858	143.4972
0.3	130.8931	179.7498
0.1	239.7242	306.6782

Table 6. Errors calculated for the spline metamodels for the routing network.

5 Conclusions

From the analysis of the results it can be concluded that the metamodels without weights result in lower errors when compared with the metamodels with weights. This means that, for both systems, using the same adjusting method, we obtained better results for the metamodels without weights.
The figure 1(a) presents the two best adjustments for the M/M/1 queueing system. The fourth degree polynomial without weights presented SSE = 5.9835 and the spline with $P = 0.999$ without weights presented SSE = 18.7992. In spite of the SSE value being lower for the polynomial fit, the spline adjustment is better, since the curve fits to the behavior of the control points. On the other hand, it is inappropriate to choose the polynomial adjustment since it presents oscillations. These oscillation result in poor metamodel predictions for input values between the design points.
For the routing network, according to the polynomial metamodels, the best fit was achieved by the fourth degree polynomial without weights. However, the values calculated for the SSE are quite high, regardless of the polynomial degree. Therefore, as in the M/M/1 queueing system, the splines allow for better adjustments in the routing network when compared with the polynomial metamodels (see figure 1(b)).



(a) M/M/1 queueing system (b) Routing network

Fig. 1. Comparison between the adjustments provided by the fourth degree polynomial and by the spline with $P = 0.999$, both without weights, for the M/M/1 queueing system (at the left) and for the routing network (at the right).

The stochastic simulation metamodels based on splines can provide better system approximations than polynomials. For both systems under study, the splines can cover the deficiencies of the polynomial metamodels, especially in the case of the routing network, where no polynomial achieved an acceptable adjustment. Based on the experimental results, the splines provide a better ability to adjust to points that follow a non-linear behavior.

References

1. Alexopoulos, C.: A comprehensive review of methods for simulation output analysis. In: Perrone, L., Wieland, F., Liu, J., Lawson, B., Nicol, D., Fujimoto, R. (eds.) Proceedings of the Winter Simulation Conference. pp. 168–178. IEEE, Piscataway, NJ (Dec 2006)
2. Banks, J., Carson, S., Nelson, B., Nicol, D.: Discrete-Event System Simulation. Prentice-Hall, Englewood Cliffs, NJ 07632, USA, 5 edn. (2009)
3. de Boor, C.: A Practical Guide to Splines. Springer Verlag, New York, USA (2001)
4. Law, A.M.: Simulation Modeling and Analysis. McGraw-Hill, New York, NY, USA, fourth edn. (2007)
5. Santos, M.I.R.: Construção de Metamodelos de Regressão Não Linear para Simulação de Acontecimentos Discretos. Ph.D. thesis, Instituto Superior Técnico, Lisboa (Dec 2002)
6. Santos, P.M., Santos, M.I.: Construction of stochastic simulation metamodels using smoothing splines. Int. J. Simulation and Process Modeling 7(4), 249–261 (2012)
7. Sargent, R.G.: Research issues in metamodeling. In: Nelson, B., Kelton, W., Clark, G. (eds.) Proceedings of the Winter Simulation Conference. pp. 888–893. IEEE, Piscataway, NJ (1991)
8. Yucesan, E.: Randomization tests for initialization bias in simulation output. Naval Research Logistics 40, 643–663 (1993)